Search Summary

Non-Patent Literature

PORTAL TO COMPUTING LITERATURE

> home > about > feedback > logout

US Patent & Trademark Office

Search Results

Search Results for: **[path profiling]**
Found **73** of **418,869 searched.**

Search within Results

GO > Advanced Search

> Search Help/Tips

Sort by: **Title** **Publication** **Publication Date** **Score** Binder

Results 1 - 20 of 73 **short listing**

Prev Page 1 2 3 4 Next Page

1 Software profiling for hot path prediction: less is more 100%
Evelyn Duesterwald , Vasanth Bala
Proceedings of the ninth international conference on Architectural support for programming languages and operating systems November 2000

2 Software profiling for hot path prediction: less is more 100%
Evelyn Duesterwald , Vasanth Bala
ACM SIGPLAN Notices November 2000
Volume 35 Issue 11

3 Efficient path profiling 100%
Thomas Ball , James R. Larus
Proceedings of the 29th annual ACM/IEEE international symposium on Microarchitecture December 1996

4 Better global scheduling using path profiles 100%
Cliff Young , Michael D. Smith
Proceedings of the 31st annual ACM/IEEE international symposium on Microarchitecture November 1998

5 Edge profiling versus path profiling: the showdown 100%
Thomas Ball , Peter Mataga , Mooly Sagiv
Proceedings of the 25th ACM SIGPLAN-SIGACT symposium on Principles of programming languages January 1998

100%

6 Improving data-flow analysis with path profiles

Glenn Ammons , James R. Larus
ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN '98 conference on Programming language design and implementation May 1998
Volume 33 Issue 5

7 An online computation of critical path profiling 99%

Jeffrey K. Hollingsworth
Proceedings of the SIGMETRICS symposium on Parallel and distributed tools January 1996

8 Exploiting hardware performance counters with flow and context sensitive profiling 99%

Glenn Ammons , Thomas Ball , James R. Larus
ACM SIGPLAN Notices , Proceedings of the 1997 ACM SIGPLAN conference on Programming language design and implementation May 1997
Volume 32 Issue 5

9 The use of program profiling for software maintenance with applications to the year 2000 problem 99%

Thomas Reps , Thomas Ball , Manuvir Das , James Larus
ACM SIGSOFT Software Engineering Notes , Proceedings of the 6th European conference held jointly with the 5th ACM SIGSOFT symposium on Software engineering November 1997
Volume 22 Issue 6

10 Resource-sensitive profile-directed data flow analysis for code optimization 98%

Rajiv Gupta , David A. Berson , Jesse Z. Fang
Proceedings of the 30th annual ACM/IEEE international symposium on Microarchitecture December 1997

11 Whole program paths 97%

James R. Larus
ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN '99 conference on Programming language design and implementation May 1999
Volume 34 Issue 5

12 Complete removal of redundant expressions 95%

Rastislav Bodík , Rajiv Gupta , Mary Lou Soffa
ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN '98 conference on Programming language design and implementation May 1998
Volume 33 Issue 5

13 Call path profiling 94%

Robert J. Hall
Proceedings of the 14th international conference on Software engineering June 1992

14 Overcoming the challenges to feedback-directed optimization (Keynote Talk) 93%

Michael D. Smith

ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN workshop on Dynamic and adaptive compilation and optimization January 2000

Volume 35 Issue 7

15 The concept of dynamic analysis 93%

Thoms Bell

ACM SIGSOFT Software Engineering Notes , Proceedings of the 7th European Engineering Conference held jointly with the 7th ACM SIGSOFT symposium on Foundations of software engineering October 1999

Volume 24 Issue 6

Dynamic analysis is the analysis of the properties of a running program. In this paper, we explore two new dynamic analyses based on program profiling:Frequency Spectrum Analysis. We show how analyzing the frequencies of program entities in a single execution can help programmers to decompose a program, identify related computations, and find computations related to specific input and output characteristics of a program.Cover ...

16 Profile assisted register allocation 88%

William C. Kreahling , Cindy Norris

Proceedings of the 2000 ACM symposium on Applied computing 2000 March 2000 2000

17 Partial method compilation using dynamic profile information 87%

John Whaley

ACM SIGPLAN Notices , Proceedings of the OOPSLA '01 conference on Object Oriented Programming Systems Languages and Applications October 2001

Volume 36 Issue 11

The traditional tradeoff when performing dynamic compilation is that of fast compilation time versus fast code performance. Most dynamic compilation systems for Java perform selective compilation and/or optimization at a method granularity. This is the not the optimal granularity level. However, compiling at a sub-method granularity is thought to be too complicated to be practical. This paper describes a straightforward technique for performing compilation and optimizations at a finer, sub-metho ...

18 A study of exception handling and its dynamic optimization in Java 87%

Takeshi Ogasawara , Hideaki Komatsu , Toshio Nakatani

ACM SIGPLAN Notices , Proceedings of the OOPSLA '01 conference on Object Oriented Programming Systems Languages and Applications October 2001

Volume 36 Issue 11

Optimizing exception handling is critical for programs that frequently throw exceptions. We observed that there are many such exception-intensive programs iin various categories of Java programs. There are two commonly used exception handling techniques, stack unwinding optimizes the normal path, while stack cutting optimizes the exception handling path. However, there has been no single exception handling technique to optimize both paths.

19 Efficient instrumentation for code coverage testing 85%

Mustafa M. Tikir , Jeffrey K. Hollingsworth

ACM SIGSOFT Software Engineering Notes , Proceedings of the international symposium on Software testing and analysis May 2002
Volume 27 Issue 3

Evaluation of Code Coverage is the problem of identifying the parts of a program that did not execute in one or more runs of a program. The traditional approach for code coverage tools is to use static code instrumentation. In this paper we present a new approach to dynamically insert and remove instrumentation code to reduce the runtime overhead of code coverage. We also explore the use of dominator tree information to reduce the number of instrumentation points needed. Our experiments show tha ...

20 Critical path analysis of TCP transactions 85%

Paul Barford , Mark Crovella
IEEE/ACM Transactions on Networking (TON) June 2001
Volume 9 Issue 3

Improving the performance of data transfers in the Internet (such as Web transfers) requires a detailed understanding of when and how delays are introduced. Unfortunately, the complexity of data transfers like those using HTTP is great enough that identifying the precise causes of delays is difficult. In this paper, we describe a method for pinpointing where delays are introduced into applications like HTTP by using critical path analysis. By constructing and profiling the ...

Results 1 - 20 of 73 **short listing**

Prev Page 1 2 3 4 Next Page

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2002 ACM, Inc.

PORTAL TO COMPUTING LITERATURE

> home > about > feedback > logout

US Patent & Trademark Office

Search Results

Search Results for: **[hierarchical path]**
Found **46** of **418,869 searched.**

Search within Results

> Advanced Search

> Search Help/Tips

Sort by: **Title** **Publication** **Publication Date** **Score** Binder

Results 1 - 20 of 46 **short listing**

Prev Page 1 2 3 Next Page

1 Restructuring for large databases: three levels of abstraction 94%
Shamkant B. Navathe , James P. Fry
ACM Transactions on Database Systems (TODS) June 1976
Volume 1 Issue 2
The development of a powerful restructuring function involves two important components—the unambiguous specification of the restructuring operations and the realization of these operations in a software system. This paper is directed to the first component in the belief that a precise specification will provide a firm foundation for the development of restructuring algorithms and, subsequently, their implementation. The paper completely defines the semantics of the restructuring of tr ...

2 Schema analysis for database restructuring 89%
Shamkant B. Navathe
ACM Transactions on Database Systems (TODS) June 1980
Volume 5 Issue 2
The problem of generalized restructuring of databases has been addressed with two limitations: first, it is assumed that the restructuring user is able to describe the source and target databases in terms of the implicit data model of a particular methodology; second, the restructuring user is faced with the task of judging the scope and applicability of the defined types of restructuring to his database implementation and then of actually specifying his restructuring needs by translating t ...

3 Hierarchical Data-Base Management: A Survey 87%
D. C. Tsichritzis , F. H. Lochovsky
ACM Computing Surveys (CSUR) January 1976
Volume 8 Issue 1

4 CONVERT: a high level translation definition language for data conversion 85%

Nan C. Shu , Barron C. Housel , Vincent Y. Lum
Communications of the ACM October 1975
Volume 18 Issue 10

This paper describes a high level and nonprocedural translation definition language, CONVERT, which provides very powerful and highly flexible data restructuring capabilities. Its design is based on the simple underlying concept of a form which enables the users to visualize the translation processes, and thus makes data translation a much simpler task. "CONVERT" has been chosen for conveying the purpose of the language and should not be confused with any other langua ...

5 Route optimization in mobile ATM networks 85%

Gopal Dommety , Malathi Veeraraghavan , Mukesh Singhal
Mobile Networks and Applications August 1998
Volume 3 Issue 2

This paper presents an algorithm for optimizing the route of a connection that becomes suboptimal due to operations such as handoffs and location-based reroutes, and applies this algorithm to the handoff management problem in mobile ATM (Asynchronous Transfer Mode) networks based on the PNNI (Private Network-to-Network Interface) standard. The route optimization algorithm uses hierarchical route information of the connection and summarized topology and loading information of the network to ...

6 DIRECT: a query facility for multiple databases 84%

Ulla Merz , Roger King
ACM Transactions on Information Systems (TOIS) October 1994
Volume 12 Issue 4

The subject of this research project is the architecture and design of a multidatabase query facility. These databases contain structured data, typical for business applications. Problems addressed are: presenting a uniform interface for retrieving data from multiple databases, providing autonomy for the component databases, and defining an architecture for semantic services.DIRECT is a query facility for heterogeneous databases. The databases and their definitions can differ in ...

7 Data conversion and restructuring: Main schema-external schema interaction in hierarchically organized data bases 82%

A. G. Dale , N. B. Dale
Proceedings of the 1977 international conference on Management of data August 1977

A class of external schemas derivable from a tree structured main schema is identified. It is shown that the properties of this class of schemas permit the construction of a processing interface such that predicates defined on an external schema can be evaluated in an occurrence structure disciplined by the main schema.

8 EXPRESS: a data EXtraction, Processing, and Restructuring System 82%

N. C. Shu , B. C. Housel , R. W. Taylor , S. P. Ghosh , V. Y. Lum
ACM Transactions on Database Systems (TODS) June 1977
Volume 2 Issue 2

EXPRESS is an experimental prototype data translation system which can access a wide variety of data and restructure it for new uses. The system is driven by two very high level nonprocedural languages: DEFINE for data description and CONVERT for data restructuring. Program generation and cooperating process techniques are used

to achieve efficient operation. This paper describes the design and implementation of EXPRESS. DEFINE and CONVERT are summarized and the implementation ar ...

9 Hierarchical optimization of optimal path finding for transportation applications 82%

Ning Jing , Yun-Wu Huang , Elke A. Rundensteiner

Proceedings of the fifth international conference on Information and knowledge management November 1996

10 A very easy hierarchical DBMS implementation 80%

Tamira Bonar , James Driscoll

Proceedings of the 1979 annual conference January 1979

The implementation of a DBMS offering a hierarchical view is described. The implementation follows a unique architecture designed to simplify DBMS implementation. The architecture incorporates basic physical storage constructs for specifying actual data storage structure, and primitive physical navigation operations for the purpose of implementing data manipulation commands. A brief discussion of the overall architecture, the physical storage language, and the physical navigation language a ...

11 Performance evaluation: Concepts of a data base simulation language 80%

Peter Scheuermann

Proceedings of the 1977 international conference on Management of data August 1977

Performance modelling of data base systems requires taking into consideration the complex interactions between the different physical design parameters and the system workload parameters. In order to facilitate a data base designer in evaluating various implementation strategies, a simulation language is presented which has three distinct components (1) data definition (2) query definition and (3) mapping to storage definition. A number of features characterize this type of descriptive mechanism ...

12 Design and evaluation of a replicated database for mobile systems 80%

S. Palazzo , A. Puliafito , M. Scarpa

Wireless Networks March 2000

Volume 6 Issue 2

The new generation mobile systems are anticipated to provide mobile users with new broadband services such as wireless multimedia, including real‐time video and high‐speed data. In these systems, the requirement on service transparency placed by the handling of mobility, both personal and terminal, implies a remarkable increase in the complexity of data management. Therefore, appropriate distributed databases (DDB) must be designed to guarantee speed in the processing an ...

13 The second Ada project: reaping the benefits 80%

Ronald L. Lawson , Mitchell L. Springer , Richard A. Howard

Proceedings of the fifth Washington Ada symposium on Ada July 1988

14 Route optimization in mobile ATM networks 80%

Gopal Dommety , Malathi Veeraraghavan , Mukesh Singhal

Proceedings of the third annual ACM/IEEE international conference on Mobile computing and networking September 1997

15 A method for monolingual thesauri merging 80%

Marios Sintichakis , Panos Constantopoulos

ACM SIGIR Forum , Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval July 1997
Volume 31 Issue SI

16 Effective graph clustering for path queries in digital map databases 80%

Yun-Wu Huang , Ning Jing , Elke A. Rundensteiner

Proceedings of the fifth international conference on Information and knowledge management November 1996

17 Browsing and querying in online documentation: a study of user interfaces and the interaction process 80%

Morten Hertzum , Erik Frøkjær

ACM Transactions on Computer-Human Interaction (TOCHI) June 1996
Volume 3 Issue 2

A user interface study concerning the usage effectiveness of selected retrieval modes was conducted using an experimental text retrieval system, TeSS, giving access to online documentation of certain programming tools. Four modes of TeSS were compared: (1) browsing, (2) conventional boolean retrieval, (3) boolean retrieval based on Venn diagrams, and (4) these three combined. Further, the modes of TeSS were compared to the use of printed manuals. The subjects observed were 87 computing new ...

18 Formal methods for evaluating information retrieval in hypertext systems 80%

Yuri Quintana , Mohamed Kamel , Rob McGeachy

Proceedings of the 11th annual international conference on Systems documentation November 1993

19 Computer mapping and geographical analysis: Some observations based on experience 77%

Robert B. Honea , Paul E. Johnson

Proceedings of the ACM 1980 annual conference January 1980

This paper focuses upon explaining six conclusions concerning what the authors consider to be an appropriate philosophy toward developing computer graphics software, particularly computer mapping. In summary, the conclusions are that computer mapping software should not be developed apart from the researcher or analyst and that those efforts developed apart tend to be esoteric in nature, Three examples involving regional energy modeling and related impact assessment are used to illustrate h ...

20 Developments in logic network path delay analysis 77%

Lionel C. Bening , Thomas A. Lane , Curtis R. Alexander , James E. Smith

Proceedings of the nineteenth design automation conference January 1982

This paper discusses path delay analysis programs as an alternative to detailed logic simulation for finding timing problems in logic networks. Fundamentals of path delay analysis are reviewed, and several previously reported methods are surveyed. This is followed by a more detailed description of a delay analysis program that we have recently implemented. Our implementation uncovers a wide variety of timing problems

and has a run time that is linearly proportional to the number of gates in ...

Results 1 - 20 of 46 **short listing**

Prev Page 1 2 3 Next Page

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2002 ACM, Inc.

PORTAL TO COMPUTING LITERATURE

> home > about > feedback > logout

US Patent & Trademark Office

Search Results

Search Results for: **[hierarchical profiling]**
Found **2** of **418,869 searched.**

Search within Results

GO > Advanced Search

> Search Help/Tips

Sort by: **Title** **Publication** **Publication Date** **Score** Binder

Results 1 - 2 of 2 **short listing**

1 Efficient and flexible location management techniques for wireless communication systems 84%

Jan Jannink , Derek Lam , Narayanan Shivakumar , Jennifer Widom , Donald C. Cox
Wireless Networks October 1997
Volume 3 Issue 5

We consider the problem of managing the information required to locate users in a wireless communication system, with a focus on designing and evaluating location management techniques that are efficient, scalable, and flexible. The three key contributions of this paper are: (1) a family of location management techniques, HiPER (for Hierarchical ProfilE Replication), that efficiently provide life-long (non-geographic) numbering with fast location lookup; (2) Pleiades, a scalable event-drive ...

2 Efficient and flexible location management techniques for wireless communication systems 80%

Jan Jannink , Derek Lam , Jennifer Widom , Donald C. Cox , Narayanan Shnivakumar
Proceedings of the second annual international conference on Mobile computing and networking November 1996

Results 1 - 2 of 2 **short listing**

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2002 ACM, Inc.

h c g e cf c

PORTAL TO COMPUTING LITERATURE

> home > about > feedback > logout

US Patent & Trademark Office

Search Results

Search Results for: **[outer path]**
Found **10** of **418,869 searched.**

Search within Results

GO > Advanced Search

> Search Help/Tips

Sort by: **Title** **Publication** **Publication Date** **Score** Binder

Results 1 - 10 of 10 **short listing**

1 Approximating shortest paths on a convex polytope in three dimensions 98%
Pankaj K. Agarwal , Sariel Har-Peled , Micha Sharir , Kasturi R. Varadarajan
Journal of the ACM (JACM) July 1997
Volume 44 Issue 4
Given a convex polytope P with n faces in R^3 , points s,t∈6P , and a parameter $0<e\leq 1$, we present an algorithm that constructs a path on 6P from s to

2 Approximating shortest paths on a convex polytope in three dimensions 97%
Sariel Har-Peled , Micha Sharir , Kasturi R. Varadarajan
Proceedings of the twelfth annual symposium on Computational geometry May 1996

3 An Efficient Implementation of Edmonds' Algorithm for Maximum Matching on Graphs 91%
Harold N. Gabow
Journal of the ACM (JACM) April 1976
Volume 23 Issue 2
A matching on a graph is a set of edges, no two of which share a vertex. A maximum matching contains the greatest number of edges possible. This paper presents an efficient implementation of Edmonds' algorithm for finding a maximum matching. The computation time is proportional to V3, where V is the number of vertices; previous implementations of Edmonds' algorithm have computation time proportional to V4

4 Approximate shortest paths and geodesic diameters on convex polytopes in three dimensions 91%
Sariel Har-Peled
Proceedings of the thirteenth annual symposium on Computational geometry August 1997

5 Constructing approximate shortest path maps in three dimensions 88%
Sariel Har-Peled
Proceedings of the fourteenth annual symposium on Computational geometry June 1998

6 An approach to the k paths problem 84%
Allen Cypher
Proceedings of the twelfth annual ACM symposium on Theory of computing April 1980
The k paths problem asks whether it is possible to find k edge-disjoint paths joining k given pairs of vertices in a graph. We show that this problem can be solved in polynomial time for k≤5 if the input graph is k+2 -connected. It follows from this result that under the same restriction, the subgraph homeomorphism problem is polynomial for any pattern graph with five or fewer edges.

7 The vertex-disjoint menger problem in planar graphs 84%
Heike Ripphausen-Lipa , Dorothea Wagner , Karsten Weihe
Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms January 1993

8 Optimal link path queries in a simple polygon 82%
Esther M. Arkin , Joseph S. B. Mitchell , Subhash Suri
Proceedings of the third annual ACM-SIAM symposium on Discrete algorithms September 1992
We develop a data structure for answering link distance queries between two arbitrary points in a simple polygon. The data structure requires O(n3) time and space for its construction and answers link distance queries in O(log n) time. Our result extends to link distance queries between pairs of segments or polygons. We also propose a simpler data structure for computing a link distance approximately, w ...

9 Effective null pointer check elimination utilizing hardware trap 77%
Motohiro Kawahito , Hideaki Komatsu , Toshio Nakatani
Proceedings of the ninth international conference on Architectural support for programming languages and operating systems November 2000

10 Effective null pointer check elimination utilizing hardware trap 77%
Motohiro Kawahito , Hideaki Komatsu , Toshio Nakatani
ACM SIGPLAN Notices November 2000
Volume 35 Issue 11

Results 1 - 10 of 10 short listing

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2002 ACM, Inc.

PORTAL TO COMPUTING LITERATURE

> home > about > feedback > logout

US Patent & Trademark Office

Search Results

Search Results for: **[profiling and counter]**
Found **1,177** of **418,869 searched.**

Warning: Maximum result set of 200 exceeded. Consider refining.

Search within Results

> Advanced Search

> Search Help/Tips

Sort by: **Title** **Publication** **Publication Date** **Score**

Results 1 - 20 of 200 **short listing**

Prev Page 1 2 3 4 5 6 7 8 9 10 Next Page

1 Optimally profiling and tracing programs 100%

Thomas Ball , James R. Larus

ACM Transactions on Programming Languages and Systems (TOPLAS) July 1994 Volume 16 Issue 4

This paper describes algorithms for inserting monitoring code to profile and trace programs. These algorithms greatly reduce the cost of measuring programs with respect to the commonly used technique of placing code in each basic block. Program profiling counts the number of times each basic block in a program executes. Instruction tracing records the sequence of basic blocks traversed in a program execution. The algorithms optimize the placement of counting/tracing code with respect to the ...

2 Rapid profiling via stratified sampling 100%

S. Subramanya Sastry , Rastislav Bodík , James E. Smith

ACM SIGARCH Computer Architecture News , Proceedings of the 28th annual international symposium on on Computer architecture May 2001 Volume 29 Issue 2

Sophisticated binary translators and dynamic optimizers demand a program profiler with low overhead, high accuracy, and the ability to collect a variety of profile types. A profiling scheme that achieves these goals is proposed. Conceptually, the hardware compresses a stream of profile data by counting identical events; the compressed profile dam is passed to software for analysis. Compressing the high-bandwidth event stream greatly reduces software overhead. Because optimizations can tole ...

3 Exploiting hardware performance counters with flow and context sensitive profiling 100%

Glenn Ammons , Thomas Ball , James R. Larus
ACM SIGPLAN Notices , Proceedings of the 1997 ACM SIGPLAN conference on Programming language design and implementation May 1997
Volume 32 Issue 5

4 Continuous profiling: where have all the cycles gone? 100%
Jennifer M. Anderson , Lance M. Berc , Jeffrey Dean , Sanjay Ghemawat , Monika R. Henzinger , Shun-Tak A. Leung , Richard L. Sites , Mark T. Vandevoorde , Carl A. Waldspurger , William E. Weihl
ACM Transactions on Computer Systems (TOCS) November 1997
Volume 15 Issue 4
This article describes the Digital Continuous Profiling Infrastructure, a sampling-based profiling system designed to run continuously on production systems. The system supports multiprocessors, works on unmodified executables, and collects profiles for entire systems, including user programs, shared libraries, and the operating system kernel. Samples are collected at a high rate (over 5200 samples/sec. per 333MHz processor), yet with low overhead (1–3% slowdown for most workloads). A ...

5 Optimally profiling and tracing programs 100%
Thomas Ball , James R. Larus
Proceedings of the 19th ACM SIGPLAN-SIGACT symposium on Principles of programming languages February 1992
This paper presents algorithms for inserting monitoring code to profile and trace programs. These algorithms greatly reduce the cost of measuring programs. Profiling counts the number of times each basic block in a program executes and has a variety of applications. Instruction traces are the basis for trace-driven simulation and analysis, and are also used in trace-driven debugging. The profiling algorithm chooses a placement of counters that is optimized—and frequently op ...

6 Continuous profiling: where have all the cycles gone? 99%
Jennifer M. Anderson , Lance M. Berc , Jeffrey Dean , Sanjay Ghemawat , Monika R. Henzinger , Shun-Tak A. Leung , Richard L. Sites , Mark T. Vandevoorde , Carl A. Waldspurger , William E. Weihl
ACM SIGOPS Operating Systems Review , Proceedings of the sixteenth ACM symposium on Operating systems principles October 1997
Volume 31 Issue 5

7 Performance analysis using the MIPS R10000 performance counters 99%
Marco Zagha , Brond Larson , Steve Turner , Marty Itzkowitz
Proceedings of the 1996 ACM/IEEE conference on Supercomputing (CDROM) November 1996
Tuning supercomputer application performance often requires analyzing the interaction of the application and the underlying architecture. In this paper, we describe support in the MIPS R10000 for non-intrusively monitoring a variety of processor events -- support that is particularly useful for characterizing the dynamic behavior of multi-level memory hierarchies, hardware-based cache coherence, and speculative execution. We first explain how performance data is collected using an integrated set ...

8 Determining average program execution times and their variance 99%
V. Sarkar
ACM SIGPLAN Notices , Proceedings of the SIGPLAN '89 Conference on

Programming language design and implementation June 1989
Volume 24 Issue 7

This paper presents a general framework for determining average program execution times and their variance, based on the program's interval structure and control dependence graph. Average execution times and variance values are computed using frequency information from an optimized counter-based execution profile of the program.

9 ProfileMe: hardware support for instruction-level profiling on out-of-order processors 99%

Jeffrey Dean , James E. Hicks , Carl A. Waldspurger , William E. Weihl , George Chrysos
Proceedings of the 30th annual ACM/IEEE international symposium on Microarchitecture December 1997

10 Reconciling responsiveness with performance in pure object-oriented languages 99%

Urs Hölzle , David Ungar
ACM Transactions on Programming Languages and Systems (TOPLAS) July 1996
Volume 18 Issue 4

Dynamically dispatched calls often limit the performance of object-oriented programs, since opject-oriented programming encourages factoring code into small, reusable units, thereby increasing the frequency of these expensive operations. Frequent calls not only slow down execution with the dispatch overhead per se, but more importantly they hinder optimization by limiting the range and effectiveness of standard global optimizations. In particular, dynamically dispatched calles prevent stand ...

11 Efficient path profiling 99%

Thomas Ball , James R. Larus
Proceedings of the 29th annual ACM/IEEE international symposium on Microarchitecture December 1996

12 Efficient instruction cache simulation and execution profiling with a threaded-code interpreter 98%

Peter S. Magnusson
Proceedings of the 29th conference on Winter simulation December 1997

13 Tracing application program execution on the Cray X-MP and Cray 2 98%

Allen D. Malony , John L. Larson , Daniel A. Reed
Proceedings of the 1990 conference on Supercomputing November 1990

Important insights into program operation can be gained by observing dynamic execution behavior. Unfortunately, many high-performance machines provide execution profile summaries as the only tool for performance investigation. We have developed a tracing library for the Cray X-MP and Cray 2 supercomputers that supports the low-overhead capture of execution events for sequential and multitasked programs. This library has been extended to use the automatic instrumentation facilities on these machi ...

14 Tracing application program execution on the Cray X-MP and Cray 2 98%

Allen D. Malony , John L. Larson , Daniel A. Reed
Proceedings of the 1990 conference on Supercomputing October 1990

Important insights into program operation can be gained by observing dynamic execution behavior. Unfortunately, many high-performance machines provide execution profile summaries as the only tool for performance investigation. We have developed a tracing library for the Cray X-MP and Cray 2 supercomputers that supports the low-overhead capture of execution events for sequential and multitasked programs. This library has been extended to use the automatic instrumentation facilities on these ...

15 Architectural and compiler support for effective instruction prefetching: a cooperative approach 97%

Chi-Keung Luk , Todd C. Mowry
ACM Transactions on Computer Systems (TOCS) February 2001
Volume 19 Issue 1

Instruction cache miss latency is becoming an increasingly important performance bottleneck, especially for commercial applications. Although instruction prefetching is an attractive technique for tolerating this latency, we find that existing prefetching schemes are insufficient for modern superscalar processors, since they fail to issue prefetches early enough (particularly for nonsequential accesses). To overcome these limitations, we propose a new instruction prefetching technique where ...

16 A portable sampling-based profiler for Java virtual machines 97%

John Whaley
Proceedings of the ACM 2000 conference on Java Grande June 2000

17 Cache decay: exploiting generational behavior to reduce cache leakage power 97%

Stefanos Kaxiras , Zhigang Hu , Margaret Martonosi
ACM SIGARCH Computer Architecture News , Proceedings of the 28th annual international symposium on on Computer architecture May 2001
Volume 29 Issue 2

Power dissipation is increasingly important in CPUs ranging from those intended for mobile use, all the way up to high-performance processors for high-end servers. While the bulk of the power dissipated is dynamic switching power, leakage power is also beginning to be a concern. Chipmakers expect that in future chip generations, leakage's proportion of total chip power will increase significantly.

This paper examines methods for reducing leakage power within the cache memori ...

18 A comparative analysis of schemes for correlated branch prediction 97%

Cliff Young , Nicolas Gloy , Michael D. Smith
ACM SIGARCH Computer Architecture News , Proceedings of the 22nd annual international symposium on Computer architecture May 1995
Volume 23 Issue 2

19 A dynamic optimization framework for a Java just-in-time compiler 97%

Toshio Suganuma , Toshiaki Yasue , Motohiro Kawahito , Hideaki Komatsu , Toshio Nakatani
ACM SIGPLAN Notices , Proceedings of the OOPSLA '01 conference on Object Oriented Programming Systems Languages and Applications October 2001
Volume 36 Issue 11

The high performance implementation of Java Virtual Machines (JVM) and just-in-

time (JIT) compilers is directed toward adaptive compilation optimizations on the basis of online runtime profile information. This paper describes the design and implementation of a dynamic optimization framework in a production-level Java JIT compiler. Our approach is to employ a mixed mode interpreter and a three level optimizing compiler, supporting quick, full, and special optimization, each of which has a differ ...

20 Storageless value prediction using prior register values 97%

Dean M. Tullsen , John S. Seng

ACM SIGARCH Computer Architecture News , Proceedings of the 26th annual international symposium on Computer architecture May 1999

Volume 27 Issue 2

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2002 ACM, Inc.

h c g e cf c

ACM PORTAL TO COMPUTING LITERATURE

> home > about > feedback > logout

US Patent & Trademark Office

Search Results

Search Results for: **[control flow graph]**
Found **862** of **418,869 searched.**

Warning: Maximum result set of 200 exceeded. Consider refining.

Search within Results

GO > Advanced Search

> Search Help/Tips

Sort by: Title Publication Publication Date Score

Results 1 - 20 of 200 **short listing**

Prev Page 1 2 3 4 5 6 7 8 9 10 Next Page

1 What's in a region?: or computing control dependence regions in near-linear time for reducible control flow 100%

Thomas Ball

ACM Letters on Programming Languages and Systems (LOPLAS) March 1993
Volume 2 Issue 1-4

Regions of control dependence identify the instructions in a program that execute under the same control conditions. They have a variety of applications in parallelizing and optimizing compilers. Two vertices in a control-flow graph (which may represent instructions or basic blocks in a program) are in the same region if they have the same set of control dependence predecessors. The common algorithm for computing regions examines each control dependence at least once. As there may be O(V x ...

2 Reverse If-Conversion 100%

Nancy J. Warter , Scott A. Mahlke , Wen-Mei W. Hwu , B. Ramakrishna Rau

ACM SIGPLAN Notices , Proceedings of the conference on Programming language design and implementation June 1993
Volume 28 Issue 6

In this paper we present a set of isomorphic control transformations that allow the compiler to apply local scheduling techniques to acyclic subgraphs of the control flow graph. Thus, the code motion complexities of global scheduling are eliminated. This approach relies on a new technique, Reverse If-Conversion (RIC), that transforms scheduled If-Converted code back to the control flow graph representation. This paper presents the predicate internal representation, the algorithms for RIC, a ...

3 Control flow graphs as a representation language 100%

Bruce A. Cota , Douglas G. Fritz , Robert G. Sargent

Proceedings of the 26th conference on Winter simulation December 1994

4 The program dependence graph and its use in optimization 100%

Jeanne Ferrante , Karl J. Ottenstein , Joe D. Warren

ACM Transactions on Programming Languages and Systems (TOPLAS) July 1987 Volume 9 Issue 3

In this paper we present an intermediate program representation, called the program dependence graph (PDG), that makes explicit both the data and control dependences for each operation in a program. Data dependences have been used to represent only the relevant data flow relationships of a program. Control dependences are introduced to analogously represent only the essential control flow relationships of a program. Control dependences are derived from the ...

5 Register allocation in structured programs 100%

Sampath Kannan , Todd Proebsting

Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms January 1995

6 Code duplication: an assist for global instruction scheduling 99%

David Bernstein , Doron Cohen , Hugo Krawczyk

Proceedings of the 24th annual international symposium on Microarchitecture September 1991

7 Logic based modeling and analysis of workflows 99%

Hasan Davulcu , Michael Kifer , C. R. Ramakrishnan , I. V. Ramakrishnan

Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems May 1998

8 The program structure tree: computing control regions in linear time 98%

Richard Johnson , David Pearson , Keshav Pingali

Proceedings of the ACM SIGPLAN '94 conference on Programming language design and implementation August 1994

In this paper, we describe the program structure tree (PST), a hierarchical representation of program structure based on single entry single exit (SESE) regions of the control flow graph. We give a linear-time algorithm for finding SESE regions and for building the PST of arbitrary control flow graphs (including irreducible ones). Next, we establish a connection between SESE regions and control dependence equivalence classes, and show how to use the algorithm to find control regions in line ...

9 Efficient instrumentation for code coverage testing 98%

Mustafa M. Tikir , Jeffrey K. Hollingsworth

ACM SIGSOFT Software Engineering Notes , Proceedings of the international symposium on Software testing and analysis May 2002 Volume 27 Issue 3

Evaluation of Code Coverage is the problem of identifying the parts of a program that did not execute in one or more runs of a program. The traditional approach for code coverage tools is to use static code instrumentation. In this paper we present a new approach to dynamically insert and remove instrumentation code to reduce the runtime overhead of code coverage. We also explore the use of dominator tree information to reduce the number of instrumentation points needed. Our experiments show tha ...

10 Dependence-based program analysis 98%

Richard Johnson , Keshav Pingali
ACM SIGPLAN Notices , Proceedings of the conference on Programming language design and implementation June 1993
Volume 28 Issue 6

Program analysis and optimization can be speeded up through the use of the dependence flow graph (DFG), a representation of program dependences which generalizes def-use chains and static single assignment (SSA) form. In this paper, we give a simple graph-theoretic description of the DFG and show how the DFG for a program can be constructed in O(EV) time. We then show how forward and backward dataflow analyses can be performed efficiently on the DFG, using ...

11 Efficiently computing static single assignment form and the control dependence graph 98%

Ron Cytron , Jeanne Ferrante , Barry K. Rosen , Mark N. Wegman , F. Kenneth Zadeck
ACM Transactions on Programming Languages and Systems (TOPLAS) October 1991
Volume 13 Issue 4

12 An overview of hierarchical control flow graph models 98%

Douglas G. Fritz , Robert G. Sargent
Proceedings of the 27th conference on Winter simulation December 1995

13 Efficiently counting program events with support for on-line queries 98%

Thomas Ball
ACM Transactions on Programming Languages and Systems (TOPLAS)
September 1994
Volume 16 Issue 5

The ability to count events in a program's execution is required by many program analysis applications. We represent an instrumentation method for efficiently counting events in a program's execution, with support for on-line queries of the event count. Event counting differs from basic block profiling in that an aggregate count of events is kept rather than a set of counters. Due to this difference, solutions to basic block profiling are not well suited to event counting. Our algorithm fin ...

14 Optimally profiling and tracing programs 98%

Thomas Ball , James R. Larus
ACM Transactions on Programming Languages and Systems (TOPLAS) July 1994
Volume 16 Issue 4

This paper describes algorithms for inserting monitoring code to profile and trace programs. These algorithms greatly reduce the cost of measuring programs with respect to the commonly used technique of placing code in each basic block. Program profiling counts the number of times each basic block in a program executes. Instruction tracing records the sequence of basic blocks traversed in a program execution. The algorithms optimize the placement of counting/tracing code with respect to the ...

15 Determining average program execution times and their variance 97%

V. Sarkar
ACM SIGPLAN Notices , Proceedings of the SIGPLAN '89 Conference on Programming language design and implementation June 1989

Volume 24 Issue 7

This paper presents a general framework for determining average program execution times and their variance, based on the program's interval structure and control dependence graph. Average execution times and variance values are computed using frequency information from an optimized counter-based execution profile of the program.

16 Extracting task-level parallelism 97%

Milind Girkar , Constantine D. Polychronopoulos

ACM Transactions on Programming Languages and Systems (TOPLAS) July 1995 Volume 17 Issue 4

Automatic detection of task-level parallelism (also referred to as functional, DAG, unstructured, or thread parallelism) at various levels of program granularity is becoming increasingly important for parallelizing and back-end compilers. Parallelizing compilers detect iteration-level or coarser granularity parallelism which is suitable for parallel computers; detection of parallelism at the statement-or operation-level is essential for most modern microprocessors, includin ...

17 Dependence flow graphs: an algebraic approach to program dependencies 97%

Keshav Pingali , Micah Beck , Richard Johnson , Mayan Moudgill , Paul Stodghill

Proceedings of the 18th ACM SIGPLAN-SIGACT symposium on Principles of programming languages January 1991

18 Interactive type analysis and extended message splitting; optimizing dynamically-typed object-oriented programs 97%

Craig Chambers , David Ungar

ACM SIGPLAN Notices , Proceedings of the conference on Programming language design and implementation June 1990 Volume 25 Issue 6

Object-oriented languages have suffered from poor performance caused by frequent and slow dynamically-bound procedure calls. The best way to speed up a procedure call is to compile it out, but dynamic binding of object-oriented procedure calls without static receiver type information precludes inlining. Iterative type analysis and extended message splitting are new compilation techniques that extract much of the necessary type information and make it possib ...

19 The implications of program dependencies for software testing, debugging, and maintenance 97%

A. Podgurski , L. Clarke

ACM SIGSOFT Software Engineering Notes , Proceedings of the ACM SIGSOFT '89 third symposium on Software testing, analysis, and verification November 1989 Volume 14 Issue 8

This paper presents a formal, general model of program dependencies. Two generalizations of control and data dependence, called weak and strong syntactic dependence, are presented. Some of the practical implications of program dependencies are determined by relating weak and strong syntactic dependence to a relation called semantic dependence. Informally, one program statement is semantically dependent on another if the latter statement can affect the execution behavior of the former. It is ...

20 Efficient and precise modeling of exceptions for the analysis of Java programs 97%

Jong-Deok Choi , David Grove , Michael Hind , Vivek Sarkar

ACM SIGSOFT Software Engineering Notes , Proceedings of the ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering

September 1999

Volume 24 Issue 5

The Factored Control Flow Graph, *FCFG,* is a novel representation of a program's intraprocedural control flow, which is designed to efficiently support the analysis of programs written in languages, such as Java, that have frequently occurring operations whose execution may result in exceptional control flow. The FCFG is more compact than traditional CFG representations for exceptional control flow, yet there is no loss of precision in using the FCFG. In this paper, we introduce the FCFG r ...

Results 1 - 20 of 200 **short listing**

Prev Page **1 2 3 4 5 6 7 8 9 10** Next Page

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2002 ACM, Inc.